Sistema de Recomendação de Filmes e Séries

ESTRUTURA DE DADOS E ALGORITMOS

Pedro Rebouças Veloso

João Filipe Dantas Cavalcanti

Ricardo César Rocha de A. C.Filho

Sobre o Sistema

O sistema consiste em um recomendador de Filmes e Séries que auxilia o usuário na escolha do que assistir. Ele oferece três formas de recomendação:

Por gênero (usuário digita o gênero)

. Seram selecionados filmes ou séries do gênero escolhido.

. Ordenados pela nota média (vote_average).

Por filme (usuário digita o nome de um filme ou série)

. Identifica os gêneros do filme ou série de referência.

. Calcula a similaridade com outros itens (mais gêneros em comum).

. Ordena primeiro pela similaridade. Em caso de empate, ordena pela nota média.

. Sugere também títulos cruzados: se o usuário escolhe um filme, mostra séries parecidas, e vice-versa.

Por exploração recursiva

. A partir de um título inicial, expande recomendações em múltiplos níveis.

. Cada título recomendado gera novas recomendações, até um limite de profundidade definido.

. Garante maior variedade e demonstra o uso de recursão no algoritmo.

Lógica e Ferramentas

. POO - Classes, Objetos,Variáveis de instância, métodos e variáveis de instância, encapsulamento com properties, diagrama de classes, herança, polimorfismo, interfaces e SOLID. Diagrama de classes atualizado, recursão, tabelas hash, grafos, pesquisa em largura, Streamlit.

. Pandas, Ast, Zipfile, Sys, abc, typing, Streamlit.

. Além da orientação a objetos e da interface gráfica. O foco foi integrar estruturas de dados eficientes e algoritmos clássicos no sistema de recomendação, tornando-o mais robusto, escalável e didático.

Tabelas Hash

. Utilizamos dicionários e conjuntos como base para armazenar relações entre gêneros.

. Essa escolha foi feita porque essas estruturas, em Python, são implementadas como tabelas hash, permitindo acesso e inserção extremamente rápidos. Dessa forma, conseguimos registrar conexões entre milhares de gêneros de filmes e séries sem comprometer a performance.

Estrutura de Grafo

. A partir das tabelas hash, construímos um grafo de coocorrência de gêneros.

. Cada gênero passou a ser representado como um nó, e cada relação entre dois gêneros se tornou uma aresta. Esse grafo possibilitou modelar como os gêneros se conectam dentro do universo de mídias, ampliando a base para recomendações mais inteligentes.

Algoritmo de Busca em Largura (BFS)

. Implementamos o algoritmo de busca em largura, que percorre o grafo explorando vizinhos em níveis de profundidade.

. Isso permitiu identificar quais gêneros estão mais próximos do escolhido pelo usuário. O resultado foi integrado ao sistema de recomendação: mídias de gêneros mais próximos passaram a ser priorizadas.

Recursão

. Introduzimos também uma função recursiva para recomendações em múltiplos níveis.

. Partindo de uma mídia inicial, a função expande recomendações em profundidade, seguindo relações de gêneros até atingir um limite definido. Isso demonstrou na prática o funcionamento de chamadas recursivas: caso base, chamadas subsequentes e controle de itens já visitados.

Diagrama de Classes

. O diagrama foi atualizado para refletir a arquitetura completa do sistema.

. Ele mostra a classe abstrata Midia e suas subclasses Filme e Serie, além da interface Recomendavel e sua implementação RecomendadorMidias. Esse diagrama tornou a estrutura mais clara, facilitando a visualização da herança, do polimorfismo e das responsabilidades de cada classe.

Revisão do SOLID

SRP (Responsabilidade Única): cada classe ficou restrita a uma única responsabilidade (modelagem, recomendação, manipulação de grafos).

OCP (Aberto/Fechado): o sistema aceita novos tipos de mídias sem modificar as classes existentes.

LSP (Substituição de Liskov): Filme e Serie podem substituir Midia sem quebrar o código.

ISP (Segregação de Interfaces): a interface Recomendavel define apenas o necessário para recomendação.

DIP (Inversão de Dependência) : a lógica trabalha contra abstrações (Midia, Recomendavel), reduzindo acoplamento.

Onde aprender: Codeacademy, DataCamp, documentacões oficias do Python e das bibliotecas

Dificuldades e Soluções

Diário de Bordo

Fonte de dados de séries: Filmes já estavam disponíveis em movies_metadata.csv. Séries exigiram busca em arquivos *IMDB (title.basics.tsv.gz, title.ratings.tsv.gz)	**Devido a isso, foi necessário fazer um tratamento de dados ausentes, parsing de colunas e filtragem por votos.**
Dificuldade na Implementação do Grafo	**A escolha foi usar dict[str, set[str]] (hash table). Garantiu eficiência, mas foi necessário revisar inserções duplicadas e normalização de strings (gêneros em minúsculo).**
Busca em Largura (BFS).Desafio: manter lista de gêneros visitados para evitar ciclos infinitos.	**Adaptamos o algoritmo clássico para limitar profundidade.**
Integração BFS ↔ Recomendações	**Foi preciso ranquear candidatos considerando distância no grafo e nota média da mídia. Critério de ordenação combinou proximidade + qualidade.**
Recursão nas Recomendações. Complexidade de evitar duplicatas em chamadas recursivas.	**Conjunto *visitados* para controle.**